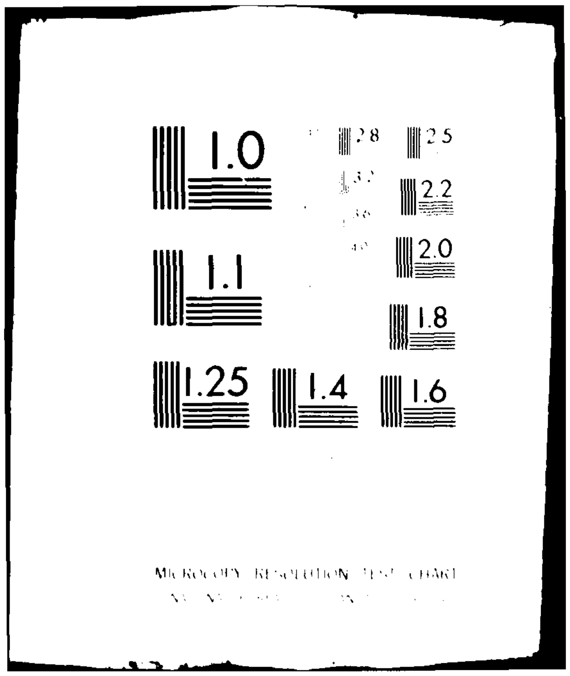
AD-A090 725   NAVAL AIR DEVELOPMENT CENTER WARMINSTER PA   SOFTWARE --ETC   F/G 9/2
DFT DECOMPOSITION AND IMPLEMENTATION FOR CONCURRENT EXECUTION O--ETC(U)
OCT 80   R S MEJZAK
UNCLASSIFIED   NADC-80110-50   NL

1 OF 1
AD A
090725

END
DATE
FILMED
11-80
DTIC


1.0
2.8
2.5
3.2
2.2
3.6
4.0
2.0
1.1
1.8
1.25
1.4
1.6
MICROCOPY RESOLUTION TEST CHART

L

(12)

REPORT NO. NADC-80110-50

DFT DECOMPOSITION AND IMPLEMENTATION FOR CONCURRENT EXECUTION ON MULTIPLE PROCESSING ELEMENTS

Richard S. Mejzak
Software and Computer Directorate
NAVAL AIR DEVELOPMENT CENTER
Warminster, Pennsylvania 18974

1 October 1980

INTERIM REPORT
AIRSASK TASK NO. A3600000/001B/0F21200000

APPROVED FOR PUBLIC RELEASE: DISTRIBUTION UNLIMITED

Prepared for
NAVAL AIR SYSTEMS COMMAND
Department of the Navy
Washington, D.C. 20361

NOTICES

REPORT NUMBERING SYSTEM - The numbering of technical project reports issued by the Naval Air Development Center is arranged for specific identification purposes. Each number consists of the Center acronym, the calendar year in which the number was assigned, the sequence number of the report within the specific calendar year, and the official 2-digit correspondence code of the Command Office or the Functional Directorate responsible for the report. For example: Report No. NADC-78015-20 indicates the fifteenth Center report for the year 1978, and prepared by the Systems Directorate. The numerical codes are as follows:

CODE	OFFICE OR DIRECTORATE
00	Commander, Naval Air Development Center
01	Technical Director, Naval Air Development Center
02	Comptroller
10	Directorate Command Projects
20	Systems Directorate
30	Sensors & Avionics Technology Directorate
40	Communication & Navigation Technology Directorate
50	Software Computer Directorate
60	Aircraft & Crew Systems Technology Directorate
70	Planning Assessment Resources
80	Engineering Support Group

PRODUCT ENDORSEMENT - The discussion or instructions concerning commercial products herein do not constitute an endorsement by the Government nor do they convey or imply the license or right to use such products.

APPROVED BY: [illegible] DATE: 5 Sept 1980

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. ~~REPORT NUMBER~~ (14) NADC-80110-50	2. GOVT ACCESSION NO. AD-A090	3. RECIPIENT'S CATALOG NUMBER 725 (9)
4. ~~TITLE (and Subtitle)~~ (6) DFT Decomposition and Implementation for Concurrent Execution on Multiple Processing Elements		5. TYPE OF ~~REPORT~~ & PERIOD COVERED Interim rept.
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) (10) Richard S. Mejzak		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Air Development Center Warminster, PA 18974		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS Air Task No. A3600000/001B/WOF21200000
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Air Development Center Warminster, PA 18974		12. ~~REPORT DATE~~ (11) 1 Oct 1980
		13. NUMBER OF PAGES
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) (12) 21		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) (16) F21200 (17) WF21200000		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report) APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Distributed Processing, Control Primitives, Semaphores, Functional Decomposition, Discrete Fourier Transform		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number)		

One of the tasks being performed for the Microprocessor Applications to Avionics project, sponsored by AIR-360B, is the development of an experimental distributed microprocessor subsystem as a vehicle to investigate distributed processing concepts at the subsystem or node level. The experimental distributed microprocessor subsystem consists of six microprocessors that employ a shared memory for communications and control. The first phase of this effort will implement and demonstrate concurrent and cooperative use of these micro-

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

S N 0102-LF-014-6601

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

313 538

~~20.~~ (Cont'd)
processors to perform a selected application function. This report defines a distributed processing concept in terms of control primitives, variables, and structures and their use in performing a decomposed DFT (Discrete Fourier Transform) application Function.

TABLE OF CONTENTS

	Page
LIST OF FIGURES	1
PURPOSE	2
EXPERIMENTAL HARDWARE CONFIGURATION	2
DFT DECOMPOSITION	2
DESIRED DEMONSTRATION OUTPUT DISPLAY	3
CONTROL STRUCTURE AND DFT IMPLEMENTATION	3
DFT APPLICATION PROGRAM DEVELOPMENT	6
COMMUNICATIONS KERNEL DEVELOPMENT	6
TESTING AND EVALUATION	6
CONCLUDING REMARKS	7
RECOMMENDATIONS	7
REFERENCES	24

LIST OF FIGURES

Figure No.	Title	Page
1	Experimental Hardware Configuration	8
2	DFT Definition	9
3	DFT Decomposition for Task wn (TSo)	10
4	DFT Decomposition for Tasks Gk(TS1) and AMP$_{(Wn)}$(TS2)	11
5	Desired Output	12
6	Control Variables	13
7	Shared Memory Data/Control Variable Lists	14
8	Slave Processor Control Primitives	15
9	Master Processor Control Primitives	16
10	Shared Memory Interactions	17
11	Master Processor Operational Scenario (Sheet 1 of 2)	18
11	Master Processor Operational Scenario (Sheet 2 of 2)	19
12	Slave Processors Operational Scenario (Sheet 1 of 4)	20
12	Slave Processors Operational Scenario (Sheet 2 of 4)	21
12	Slave Processors Operational Scenario (Sheet 3 of 4)	22
12	Slave Processors Operational Scenario (Sheet 4 of 4)	23

PURPOSE

This report describes an approach for designing a control structure together with decomposing and implementing a DFT (Discrete Fourier Transform) for concurrent execution on multiple processing elements. It is intended to serve as a working document for application program development at NAVAIRDEVCEN and for guiding the development of a communications kernel by JPL (Jet Propulsion Laboratory, Pasadena, CA). The DFT was chosen as an experimental application to investigate distributed processing concepts because of its highly regular and decomposable structure for concurrent execution. Also contained herein is a description of the demonstration that will be performed at the completion of the initial phase of the distributed processing task.

EXPERIMENTAL HARDWARE CONFIGURATION

The hardware configuration consists of IMSAI 8080 chassis which are independent, 8-bit microcomputer units. These chassis are linked together to form a multiple processing system by means of a shared memory facility. This facility consists of hardware which provides a bus structure to enable up to six microcomputers to be interconnected. It provides polling and arbitration logic so that only one processor has access to shared memory at any one time. The experimental configuration to be used for the DFT implementation is shown in figure 1. For discussion purposes, five of the processors are designated as slaves and one as a master. In actual operation, the slave processors will be cooperating to compute the DFT where the master will provide external input, output, and control functions. With this implementation, commands to perform a DFT iteration will be provided through the master. It may be noted that although five slave processors are shown in figure 1, the control structure described herein can operate with a variable number of processors.

DFT DECOMPOSITION

The definition of a DFT is provided in figure 2 where the solution results in amplitude values for a specified number of frequency points. It may be noted that if the number of points are set to 2^{α} where α is an integer greater than zero, then advantages of symmetry can be taken in terms of complex conjugates to reduce the number of computations. However, this symmetry will not be taken into account for this implementation.

The solution of the DFT can be approached several ways. To provide for more shared memory interactions, an approach which results in three separate tasks was selected. The first task requires the computation of frequency terms for each row as shown in figure 3. The second task requires the computation of real and imaginary terms for each column as shown in figure 4a. The third task requires the solution of amplitude terms for each row as shown in figure 4b. These tasks are defined in terms of input-process-output to indicate shared memory accesses for input and output and processing that is performed by means of programs in local memory. Each of the three tasks are comprised of N=K subtasks where N is the number of frequency points specified. These subtasks are executed concurrently by the slave processors, see figures 3 and 4.

DESIRED DEMONSTRATION OUTPUT DISPLAY

To determine the manner in which the slave processors cooperate to perform a DFT iteration, certain variables are maintained in shared memory for each of the slave processors. Each slave processor maintains a record of the number of subtasks that are performed for each task by means of these variables. The slave processors also update a cumulative counter to indicate if all the required subtasks are completed for each task. These variables are detailed in subsequent sections of this report and will also be useful for future fault-tolerant studies where the loss of processors due to failure can readily be observed.

Figure 5a depicts the output that is desirable for demonstration purposes. It contains the previously described table of variables in addition to a frequency/amplitude plot resulting from the current DFT iteration based on the assumed input function in figure 5b. This output format will be displayed on a model 1620 Diablo printer which has plotting capability.

CONTROL STRUCTURE AND DFT IMPLEMENTATION

The following paragraphs provide a detailed description of the concept selected for implementation in terms of shared memory variables and formats, control primitives, shared memory interactions, and processor operational scenarios.

SHARED MEMORY VARIABLES AND FORMATS

Shared memory variables are defined in terms of control and data words. Control variables are provided for resolving shared memory access rights, task assignments, and iteration completions. Use of the control variables is not unique to the DFT implementation since this is a generalized technique that can be used for other applications. Data variables are parameters used in performing the computational tasks and are application specific.

Control

Three control variables have been identified as being required to control processor interactions, see figure 6. The key variable is the BI/TLP which is a combination semaphore and task list pointer combined in one byte (8 bits). The BI is a 2-bit semaphore that controls all access rights to shared memory for the conditions shown in figure 6. The TLP is a 6-bit task list pointer that specifies the current task to be performed. This is the only control variable that must be located in a special area of memory called the Flag Block. Being located in the Flag Block guarantees uninterrupted access to shared memory for a time period that is hardware settable. The BI/TLP must be accessible by all processors.

TSs is an 8-bit word that is used to associate specific data variables with a subtask of a task pointed to by the TLP. This variable does not have to be located in the Flag Block but must be accessible by all processors. One TS is required for every task, i.e., every task pointed to by the TLP.

CTCt is an 8-bit cumulative task counter. It is used to count the number of subtasks performed for each task in order to ensure completion of a task. This variable does not have to be located in the Flag Block but must be accessible by all processors. One CTC is required for every task, i.e., every task pointed to by the TLP.

Data

Data variables are parameters that are unique to the particular application. Figure 7 summarizes both the data and control variables required to implement the DFT together with their required formats. It may be noted that shared memory storage requirements can be estimated from figure 7 based on the number of points used for the DFT. However, a constraint that is currently inherent in this implementation is that the integer formats for the TC, TS, and CTC for the data and control variables do not permit a DFT over 128 points to be computed if the number of points is based on 2^{α}. This constraint is removed if these word lengths are extended or converted to floating point. For purposes of this experiment however, 128 points are sufficient.

It may be noted that the data variables need not be located in the Flag Block of shared memory, but all variables must be accessible by all processors.

CONTROL PRIMITIVES

Control primitives are operations which give each process a means to guarantee mutually exclusive access to data shared among processes. Two primitives have been identified to control process interactions and involve operations on the BI semaphore. These primitives are similarly implemented for both the master and slave processors but perform different operations on the BI semaphore.

Slave Processors

In order to obtain exclusive access to shared memory, a slave processor employs a SEIZEs primitive as shown in figure 8. At the completion of the seize execution, other processors can access shared memory but no variables can be disturbed until the currently seizing processor executes a RELEASEs primitive as shown in figure 8. The time delays shown are provided so that the lockout period Δt can expire prior to performing another immediate seize attempt. If the software time delay were not provided and the same processor attempted another immediate seize upon failure to do so the first time, would result in an insufficient lockout time.

Master Processor

The master processor employs the same primitive techniques as used for the slave processors but imposes different conditions by means of the BI semaphore. These primitives are shown in figure 9.

SHARED MEMORY INTERACTIONS

Figure 10 is provided to illustrate interactions in shared memory with respect to the control variables TLP and TS and the DFT data variables. As shown therein, the TPL points to one of three tasks designated by TS. The TS in turn points to the appropriate location in shared memory where data is to be either fetched or stored. When the TLP reaches the value of 4, one DFT iteration is completed.

OPERATIONAL SCENARIO

The concepts developed in previous paragraphs can now be combined in an operational scenario for implementing the DFT application. Two scenarios are developed, i.e., for the master and slave processors.

Master Processor

An operational scenario for the master processor is depicted in figure 11. The assumptions imposed are:

- Operational programs are already loaded into RAM or are resident in EPROM
- Operational programs unique to the master are stored in local memory
- On power up, the slave processors execute a software time delay until the master can initialize the BI semaphore

It may be noted that a time delay equal to at least five times the lockout period, Δt , must be executed by each of the slave processors. This is required since the master may be the last processor to be polled.

Once shared memory is seized by the master processor, external control is provided by means of a terminal. Prior to releasing shared memory to the slave processors the variables shown in figure 11 must be initialized. This can either be done manually via the keyboard or programmatically from variables resident on EPROM. After initializing the variables, shared memory can be released to the slave processors by setting the BI/TLP control word to $(C1)_{16}$ as shown in figure 9. While the slave processors are performing a DFT iteration, the master processor looks for a completion indicator via the BI semaphore each time it is polled. As soon as one DFT iteration is completed, the master processor immediately displays a completion indicator. The master processor then proceeds to read in required variables into local memory for conversion and display purposes. At this point, no further action, other than checking the BI semaphore by the slave processors when polled, is taken until initiated by the master, see figure 11.

Slave Processors

An operational scenario for the slave processors is depicted in Figure 12.

The assumptions imposed are:

- Operational programs are already loaded into RAM or are resident in EPROM
- Operational programs are stored in local memory and are the same for all slave processors
- On power up, all slave processors execute a software time delay equal to at least five times the lockout period, Δt

Slave processor operations are initiated after the time delay where the variable N is set equal to zero. Data is then fetched and tasks processed as directed by the TLP and TS. It may be noted that since some tasks may depend on results from previous tasks, no subsequent task is permitted to initiate until all results of the current task are stored in shared memory. This is ensured by the CTC control variable, see figure 12. The TLP and TS pointers are incremented as required by the slave processors as well as the TC and CTC. A DFT iteration is considered completed when the TLP is equal to 4 and the CTC AMPn is equal to N at which time control is returned to the master processor by means of the BI semaphore.

DFT APPLICATION PROGRAM DEVELOPMENT

Application program development is currently being performed at NAVAIRDEVCEN in 8080 assembly language. Many of the floating point subroutines necessary to implement the slave processor tasks as defined herein have already been developed and tested; however, they must be reassembled for relocation purposes. The individual computational tasks can be developed in a modular fashion by employing these floating point subroutines. Other application development tasks required include special conversion and display routines for the master processor.

COMMUNICATIONS KERNEL DEVELOPMENT

The control aspects of the selected implementation need to be considered by JPL in their development of a communications kernel. It is required that the communications kernel accommodate the control structure and application function defined herein.

TESTING AND EVALUATION

It is expected that this concept will be tested and demonstrated on the laboratory model by the end of FY 80 or early FY 81. Future evaluations will concentrate on areas such as performance comparisons based on varying the number of processors and bus contention factors as a function of local processing and common data base access times. The results of these evaluations will be published in a future report.

For the next phase of this task, fault tolerant investigations will be performed to identify techniques that can be directly implemented and evaluated on the baseline laboratory model. Fault tolerant studies will concentrate on three general areas, i.e., processors, bus, and shared memory. Techniques investigated will concentrate on optimizing the use of the available processors and existing control structure.

CONCLUDING REMARKS

Distributed processing is a state-of-the-art technology and currently lacks formalism with respect to implementation techniques. This report provides an approach that forms a baseline for future investigations.

The approach defined herein assumes interprocessor communications to be annonymous. In this scheme, all processors access the entire common database by using control primitives. Ownership of a selected area within the common database is momentary and is enforced by a lock and is determined by task and subtask pointers.

This approach can be contrasted to one in which processor to processor communications is explicit. This scheme requires control primitives to know which processor is to receive or send data to (or from) another processor. Access of a common database is restricted to partitioned mailboxes determined at system generation time which resemble dedicated hardware channels.

In summary, the approach described herein was chosen because it was felt that fault tolerant schemes could more readily be investigated This is due to the fact that loss of processors do not affect the control structure.

RECOMMENDATIONS

To provide a baseline for further distributed concept investigations, it is recommended that the control structure and DFT application function, described herein, be implemented and demonstrated on the existing 8080 microprocessor laboratory model at NAVAIRDEVCEN. It may be noted that, although 8080 microprocessors are being employed because of their immediate availability, this concept is applicable to any microprocessor.

Follow-up evaluations and studies recommended for the baseline model include:

- Performance comparisons based on varying the number of processors and bus contention evaluations based on local processing and shared memory access times.

- Fault tolerant studies to identify techniques for circumventing processor, bus, and shared memory failures. Candidate techniques should be capable of being directly implemented and evaluated on the laboratory model.

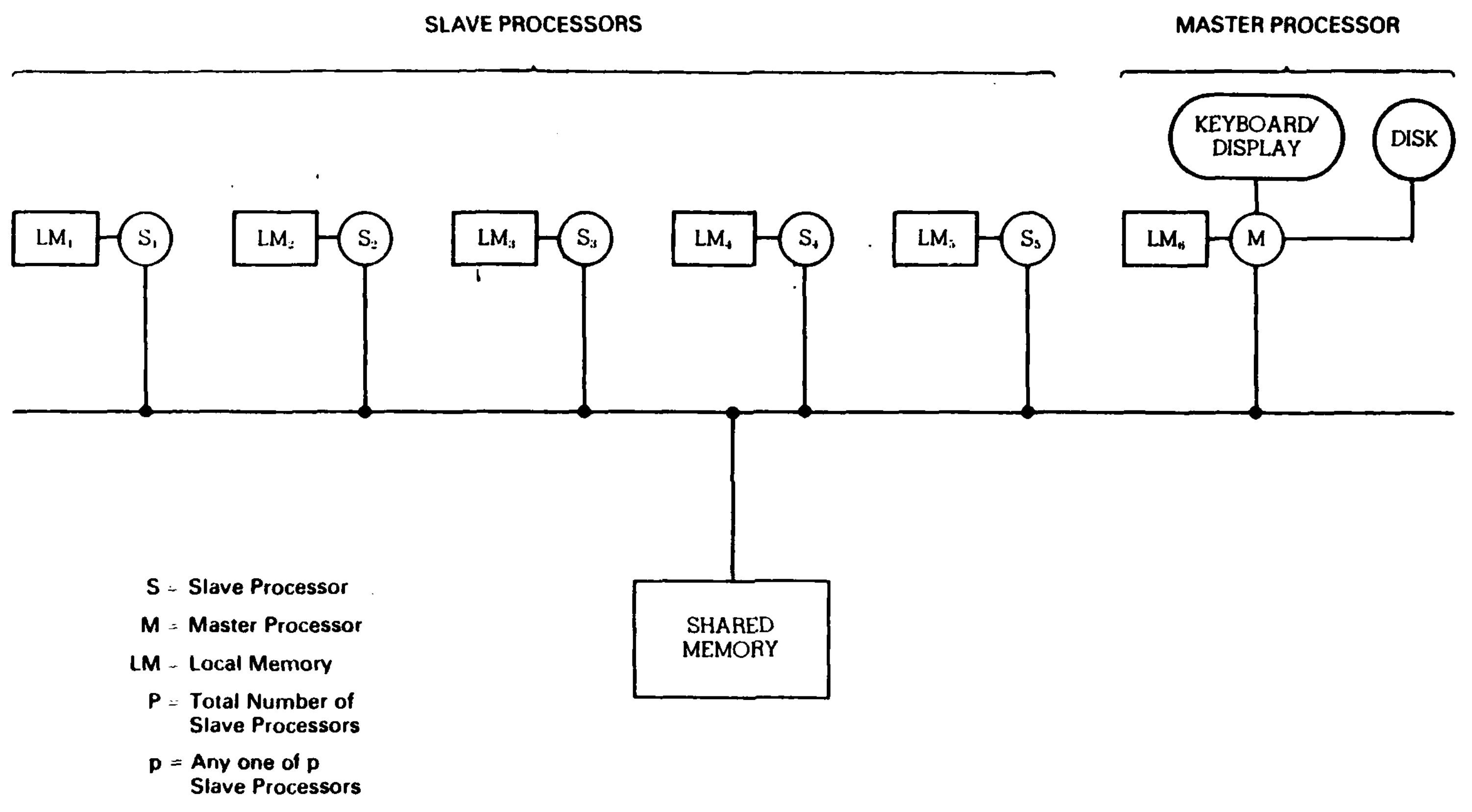


Figure 1. Experimental Hardware Configuration

A DFT can be defined in the following matrix form:

$$G = WF$$

If we let:

- $n = 0, 1, 2, \ldots, N - 1$ = matrix row number and frequency step
- $k = 0, 1, 2, \ldots, K - 1$ = matrix column number and time step
- $N = K$ but maintaining n and k notations to distinguish rows from *columns*

Then:

- W is an $N \times K$ matrix consisting of the terms

$$W^{n,k} = e^{(-2\pi j/N)(nk \bmod N)}$$
$$= \mathrm{COS}\left[\left(\frac{2\pi}{N}\right)\left(nk \bmod N\right)\right] - j\,\mathrm{SIN}\left[\left(\frac{2\pi}{N}\right)\left(nk \bmod N\right)\right]$$

- F is a $K \times 1$ matrix representing the function $F_{(tk)}\,{}^{T}/2\pi K$ over the time span T
- G is an $N \times 1$ matrix where $G_n = {}^{T}/2\pi K \sum_{k=0}^{k-1} W^{n,k} F_{(tk)}$

In expanded form, $G = WF$ can be written as:

$$\begin{pmatrix} G_0 \\ G_1 \\ G_2 \\ \vdots \\ G_{N-1} \end{pmatrix} = \begin{pmatrix} W^{0,0} & W^{0,1} & W^{0,2} & \cdots W^{0,k-1} \\ W^{1,0} & W^{1,1} & W^{1,2} & \cdots W^{1,k-1} \\ W^{2,0} & W^{2,1} & W^{2,2} & \cdots W^{2,k-1} \\ \vdots & & & \vdots \\ W^{N-1,0} & W^{N-1,1} & W^{N-1,2} & W^{N-1,k-1} \end{pmatrix} \begin{pmatrix} F_{t0}\,{}^{T}/2\pi K \\ F_{t1}\,{}^{T}/2\pi K \\ F_{t2}\,{}^{T}/2\pi K \\ \vdots \\ F_{tk-1}\,{}^{T}/2\pi K \end{pmatrix}$$

Since:

$$G_{n,k}\ (\mathrm{Real}) = \left\{\mathrm{COS}\left[\left(\frac{2\pi}{N}\right)\left(nk \bmod N\right)\right]\right\} F_{(tk)}\,{}^{T}/2\pi K$$

$$G_{n,k}\ (\mathrm{Imaginary}) = -\,j\left\{\mathrm{SIN}\left[\left(\frac{2\pi}{N}\right)\left(nk \bmod N\right)\right]\right\} F_{(tk)}\,{}^{T}/2\pi K$$

$$W_n = n\Delta w \text{ where } \Delta w = \frac{2\pi k}{NT}$$

$$\left|G_n\right| = \sqrt{\left[\sum_{k=0}^{k-1} G_{n,k}\ (\mathrm{Real})\right]^2 + \left[\sum_{k=0}^{k-1} G_{n,k}\ (\mathrm{Imaginary})\right]^2}$$

Then amplitude/frequency values can be obtained as follows:

$$\mathrm{AMP}_{(W_n)} = \Delta w \left|G_n\right|$$

Figure 2. DFT Definition

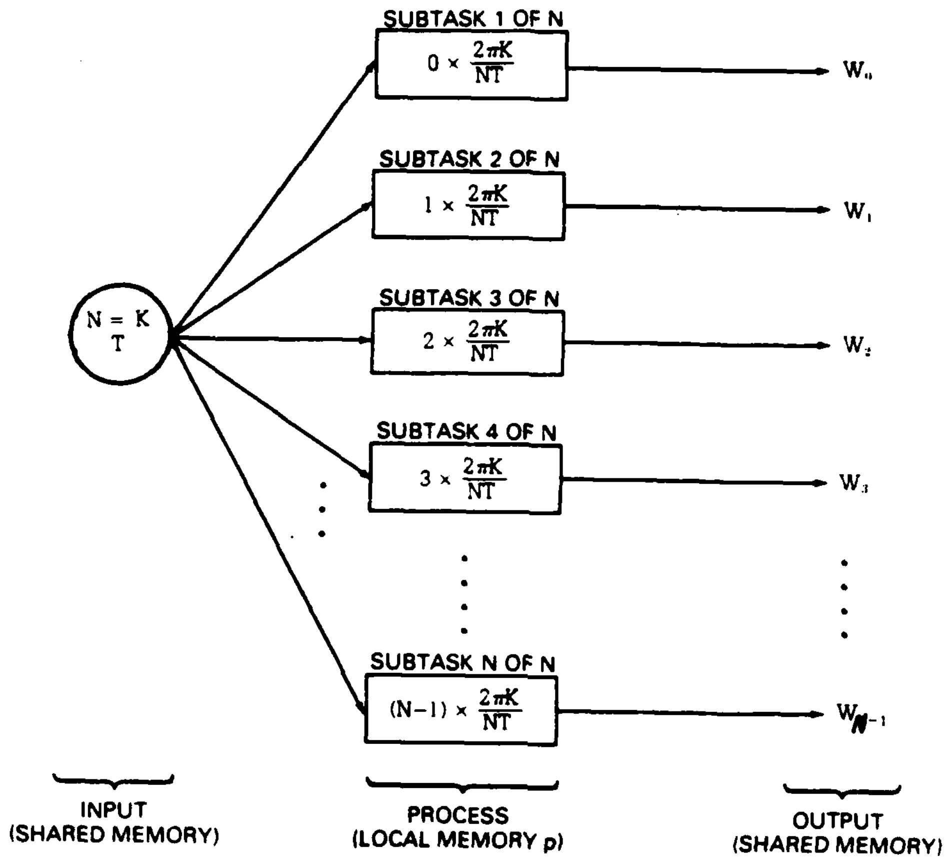
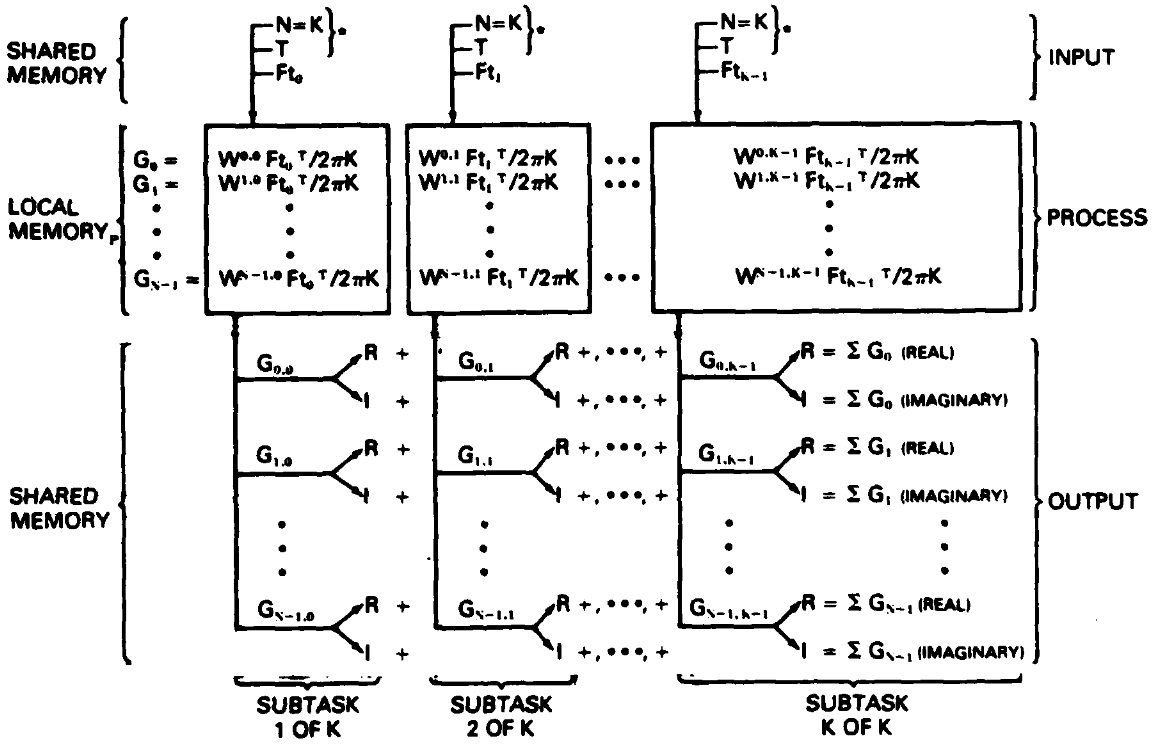
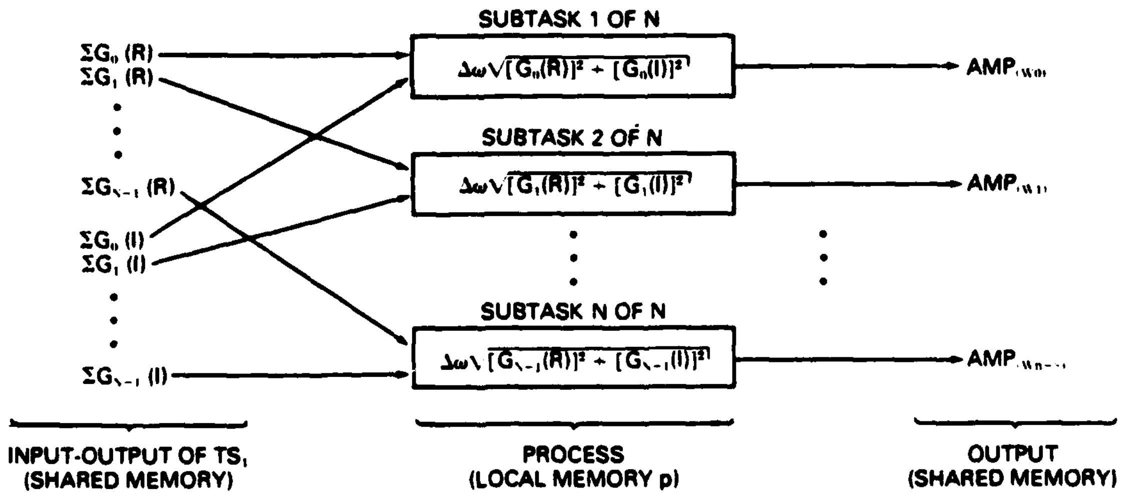


Figure 3. DFT Decomposition for Task wn (TSo)



*VALUES ALREADY FETCHED FOR TS_0

a. Task Gk (TS_1)

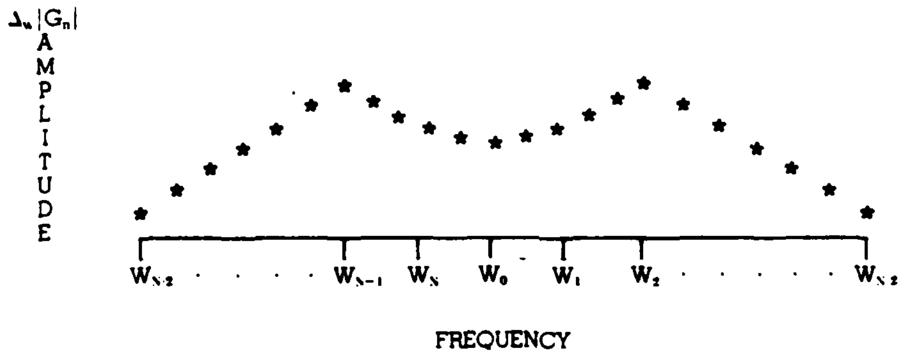


b. TASK $AMP_{(W_n)}$ (TS_2)

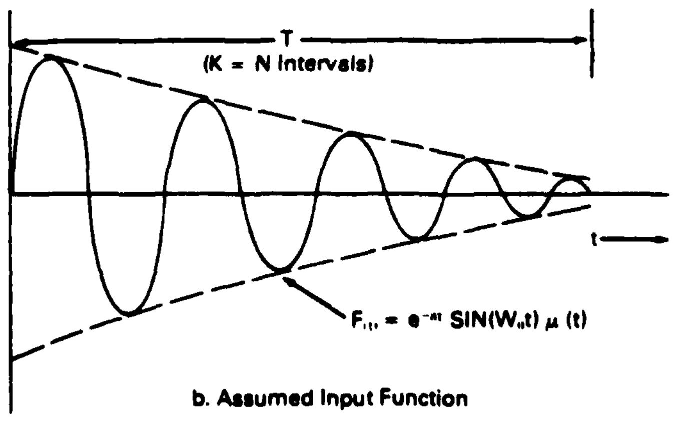
Figure 4. DFT Decomposition for Tasks Gk(TS1) and $AMP_{(W_n)}$ (TS2)

SLAVE PROCESSOR	TASK/SUBTASK COMPLETIONS		
	W_n	G_k	AMP_n
1	X	X	X
2	X	X	X
3	X	X	X
4	X	X	X
5	X	X	X
CTC*	X	X	X

*Cumulative Task Counter



a. Output Display



b. Assumed Input Function

Figure 5. Desired Output

- BI/TLP_t

—FORMAT: $\underbrace{\overset{M_{SB}}{\boxed{7}}\boxed{6}}_{BI}\underbrace{\boxed{5}\boxed{4}\boxed{3}\boxed{2}\boxed{1}\overset{L_{SB}}{\boxed{0}}}_{TLP} \longleftarrow$ BIT POSITION (ONE BYTE)

—BI is a 2-bit semaphore and indicates the following conditions:

SEMAPHORE		CONDITION
B	I	
0	0	SHARED MEMORY BLOCKED AND ITERATION COMPLETED
0	1	SHARED MEMORY BLOCKED AND ITERATION IN PROGRESS
1	0	SHARED MEMORY AVAILABLE AND ITERATION COMPLETED
1	1	SHARED MEMORY AVAILABLE AND ITERATION IN PROGRESS

—*TLP is a 6-bit Task List Pointer that can point to any one of 64 tasks,* i.e., TS_0 through TS_{63}.

- TS_t

—FORMAT: $\overset{M_{SB}}{\boxed{7}}\boxed{6}\boxed{5}\boxed{4}\boxed{3}\boxed{2}\boxed{1}\overset{L_{SB}}{\boxed{0}} \longleftarrow$ BIT POSITION (ONE BYTE)

—TS_t is an 8-bit word used to associate corresponding data with a task and can take on 256 values, i.e., TS=0 through TS=255.

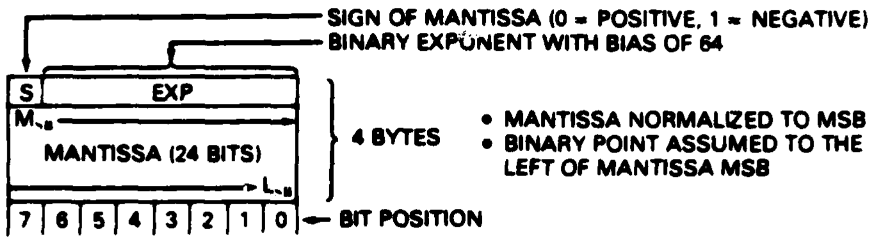
- CTC_t

—FORMAT: $\overset{M_{SB}}{\boxed{7}}\boxed{6}\boxed{5}\boxed{4}\boxed{3}\boxed{2}\boxed{1}\overset{L_{SB}}{\boxed{0}} \longleftarrow$ BIT POSITION (ONE BYTE)

—CTC is an 8-bit Cumulative Task Counter. One CTC is required for each type of task being performed, i.e., the number of CTC's are equal to the number of tasks pointed to by TLP.

Figure 6. Control Variables

- INTEGER FORMAT (I): | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | BIT POSITION (ONE BYTE)
- FLOATING POINT WORD FORMAT (FP):


SIGN OF MANTISSA (0 = POSITIVE, 1 = NEGATIVE)
BINARY EXPONENT WITH BIAS OF 64
S
EXP
M_{SB}
MANTISSA (24 BITS)
L_{SB}
4 BYTES
7 6 5 4 3 2 1 0
BIT POSITION

- MANTISSA NORMALIZED TO MSB
- BINARY POINT ASSUMED TO THE LEFT OF MANTISSA MSB

- DATA LISTS

DATA TYPE	CONTENTS	NUMBER OF WORDS	FORMAT
N = K	NUMBER OF FREQUENCY STEPS N AND NUMBER OF TIME STEPS K	1 (4 BYTES)	FP
T	$F_{(t)}$ TIME SPAN	1 (4 BYTES)	FP
$F_{(tk)}$	$F_{(t)}$ VALUES OVER TIME SPAN T	K (K×4 BYTES)	FP
ΣG_n(REAL)	SUM OF REAL VALUES OF G FOR EACH OF N ROWS	N (N×4 BYTES)	FP
ΣG_n(IMAG)	SUM OF IMAGINARY VALUES OF G FOR EACH OF N ROWS	N (N×4 BYTES)	FP
AMP_n	AMPLITUDE VALUES CORRESPONDING TO N W_n VALUES	N (N×4 BYTES)	FP
W_n	FREQUENCY VALUES CORRE-SPONDING TO N AMP_n VALUES	N (N×4 BYTES)	FP
TC_{Wn}	TASK COUNTER W_n FOR EACH OF P SLAVE PROCESSORS	P* (P BYTES)	I
TC_{Gk}	TASK COUNTER G_k FOR EACH OF P SLAVE PROCESSORS	P* (P BYTES)	I
TC_{AMPn}	TASK COUNTER AMP_n FOR EACH OF P SLAVE PROCESSORS	P* (P BYTES)	I

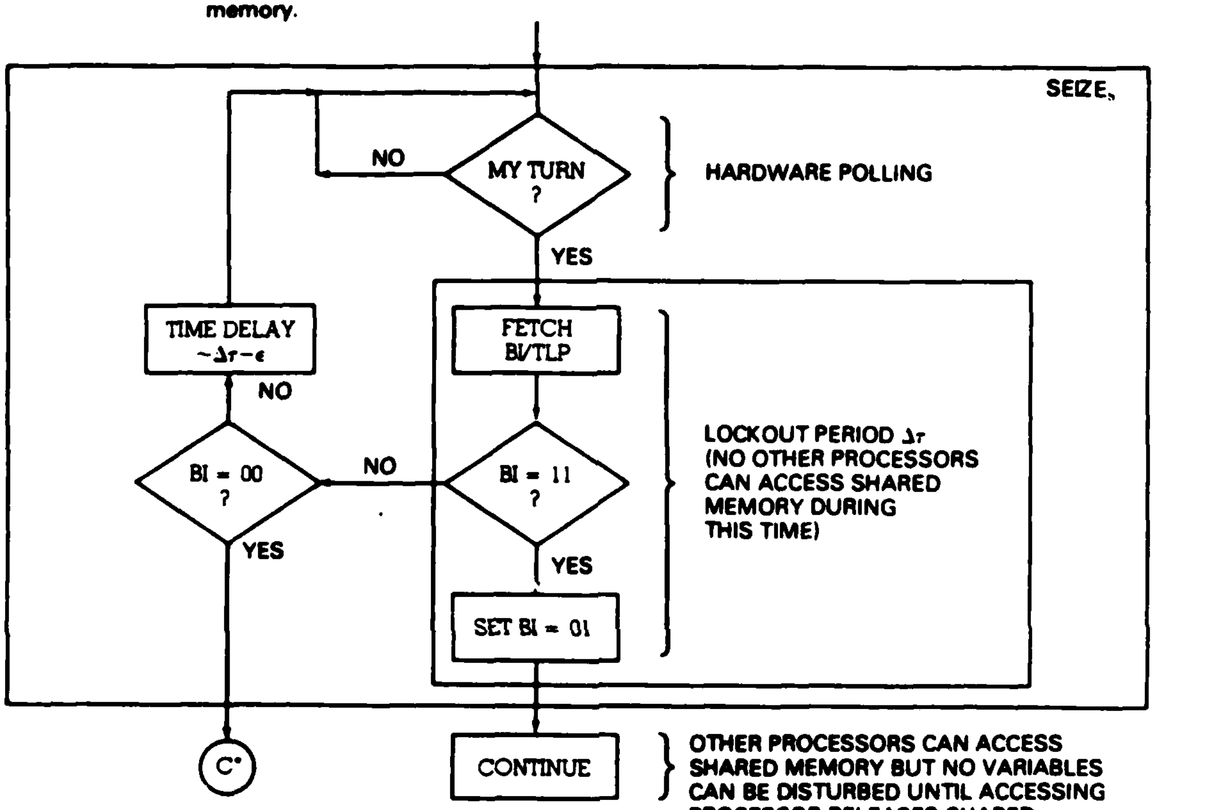
*ONE FOR EACH SLAVE PROCESSOR p

- CONTROL VARIABLE LIST

CONTROL VARIABLE	CONTENTS	NUMBER OF WORDS	FORMAT
BI/TLP	CONTROL SEMAPHORE AND TASK LIST POINTER	1 (1 BYTE)	I
TS	SLAVE PROCESSOR TASK DATA POINTERS	T (1 BYTE FOR EACH TASK POINTED TO BY TLP)	I
CTC	CUMULATIVE TASK COUNTERS	T (1 BYTE FOR EACH TASK POINTED TO BY TLP)	I

Figure 7. Shared Memory Data/Control Variable Lists

- **SEIZE$_S$ PRIMITIVE**
  This primitive is executed by the slave processors when accessing shared memory.


SEIZE$_S$
NO
MY TURN ?
HARDWARE POLLING
YES
TIME DELAY ~Δτ−ε
FETCH BI/TLP
NO
BI = 00 ?
NO
BI = 11 ?
LOCKOUT PERIOD Δτ (NO OTHER PROCESSORS CAN ACCESS SHARED MEMORY DURING THIS TIME)
YES
YES
SET BI = 01
C*
CONTINUE
OTHER PROCESSORS CAN ACCESS SHARED MEMORY BUT NO VARIABLES CAN BE DISTURBED UNTIL ACCESSING PROCESSOR RELEASES SHARED MEMORY BY MEANS OF THE RELEASE$_S$ PRIMITIVE

*SEE FIG. 12 (Sheet 1 of 4)

- **RELEASE$_S$ PRIMITIVE**
  This primitive is executed by slave processors when releasing shared memory.

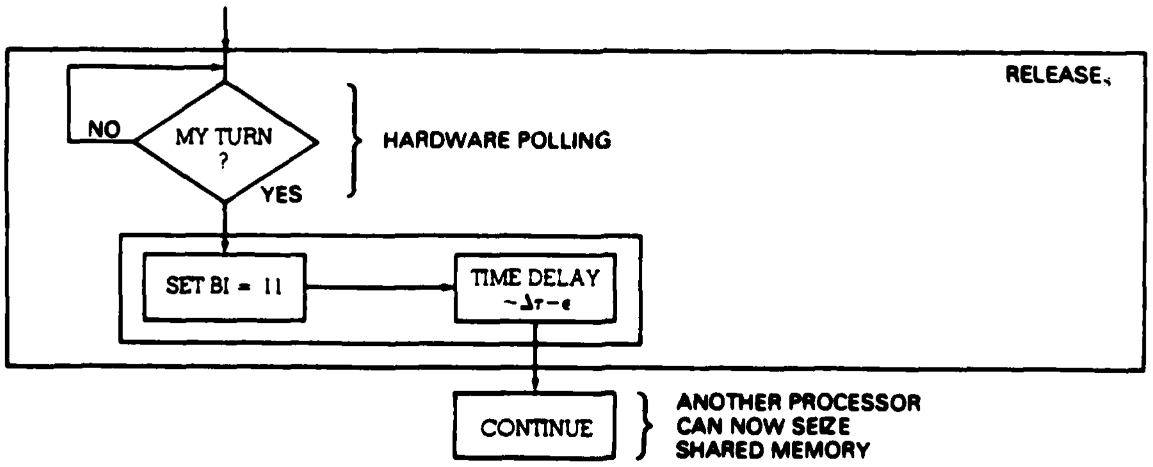
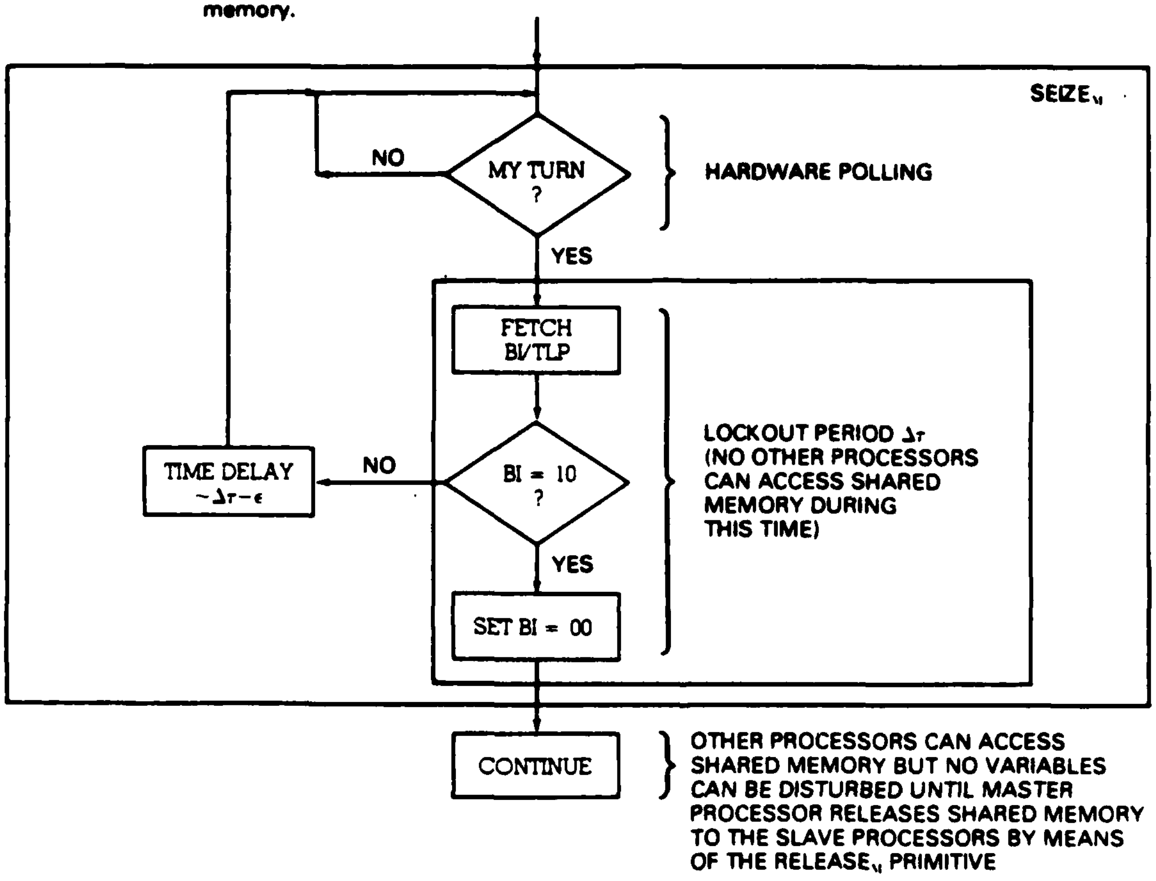


Figure 8. Slave Processor Control Primitives

- SEIZE$_M$ PRIMITIVE
  This primitive is executed by the master processor when accessing shared memory.


SEIZE$_M$
NO
MY TURN ?
HARDWARE POLLING
YES
FETCH BI/TLP
TIME DELAY $\sim\Delta\tau-\epsilon$
NO
BI = 10 ?
LOCKOUT PERIOD $\Delta\tau$ (NO OTHER PROCESSORS CAN ACCESS SHARED MEMORY DURING THIS TIME)
YES
SET BI = 00
CONTINUE
OTHER PROCESSORS CAN ACCESS SHARED MEMORY BUT NO VARIABLES CAN BE DISTURBED UNTIL MASTER PROCESSOR RELEASES SHARED MEMORY TO THE SLAVE PROCESSORS BY MEANS OF THE RELEASE$_M$ PRIMITIVE

- RELEASE$_M$ PRIMITIVE
  This primitive is executed by the master processors when releasing shared memory to the slave processors for performing an iteration.

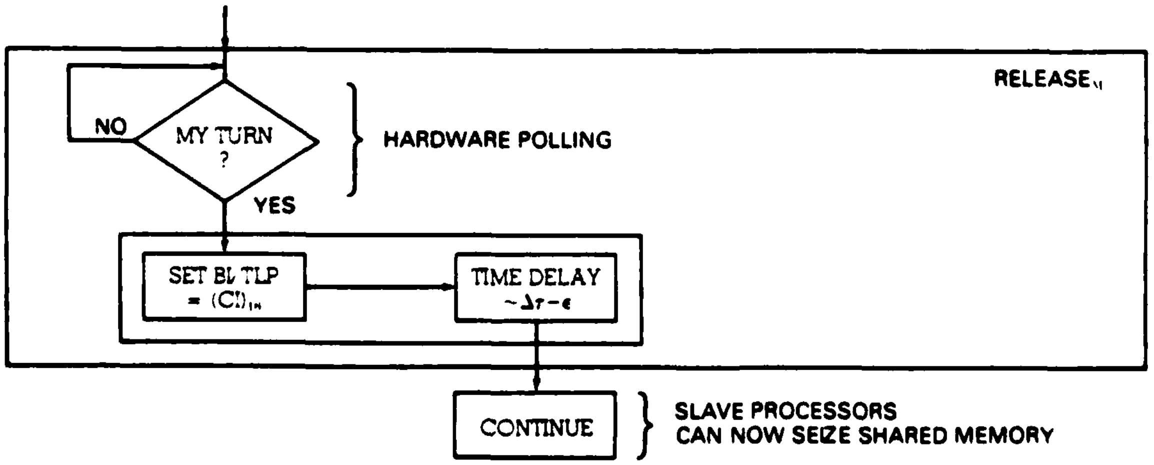

RELEASE$_M$
NO
MY TURN ?
HARDWARE POLLING
YES
SET BI/TLP = $(CI)_{1N}$
TIME DELAY $\sim\Delta\tau-\epsilon$
CONTINUE
SLAVE PROCESSORS CAN NOW SEIZE SHARED MEMORY

Figure 9. Master Processor Control Primitives

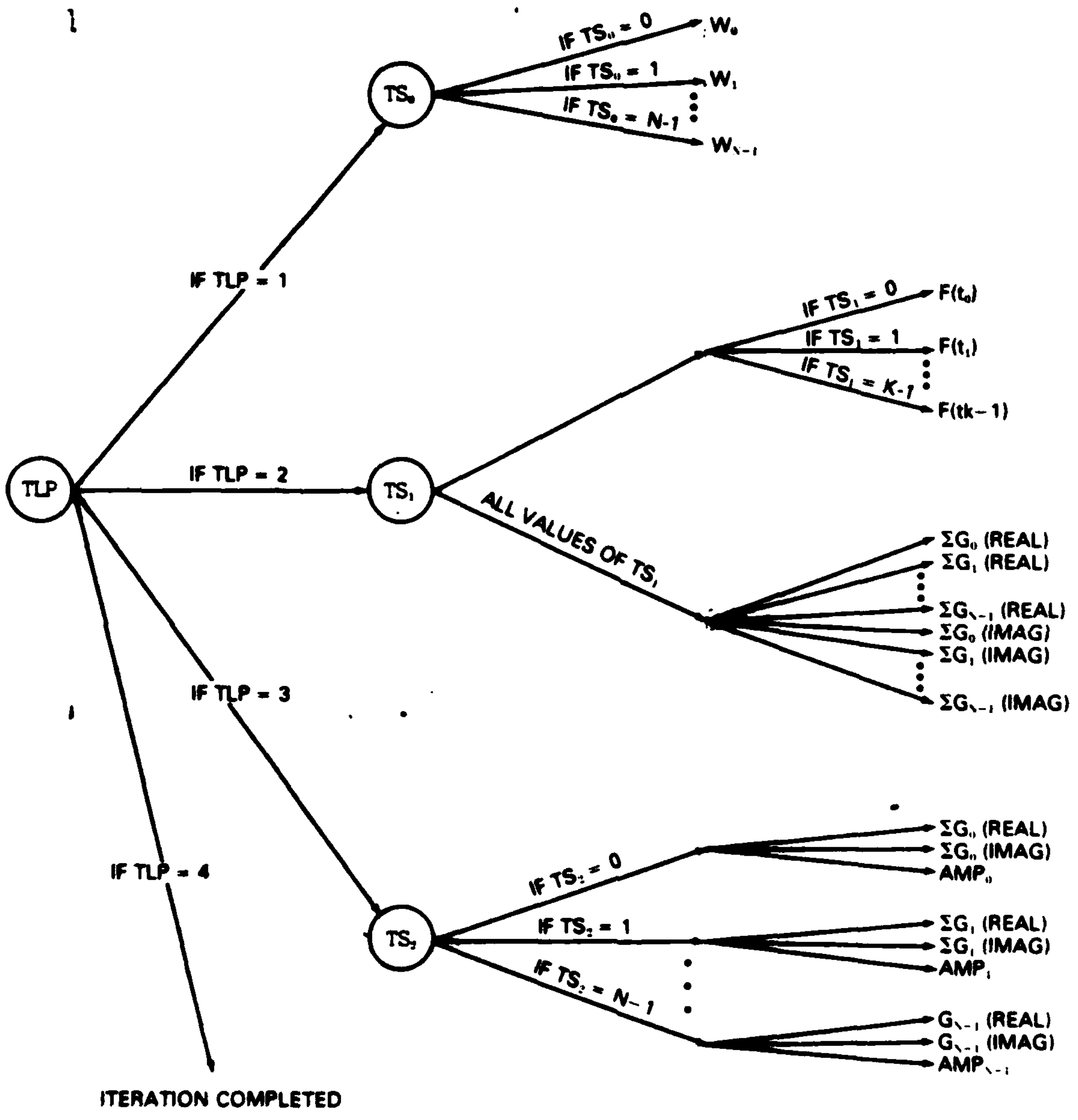


Figure 10. Shared Memory Interactions

ASSUMPTIONS:

- Operational programs already loaded into RAM or resident in EPROM
- Master processor programs stored in local memory
- On power up, slave processors execute time delay software until master processor can initialize BI semaphore

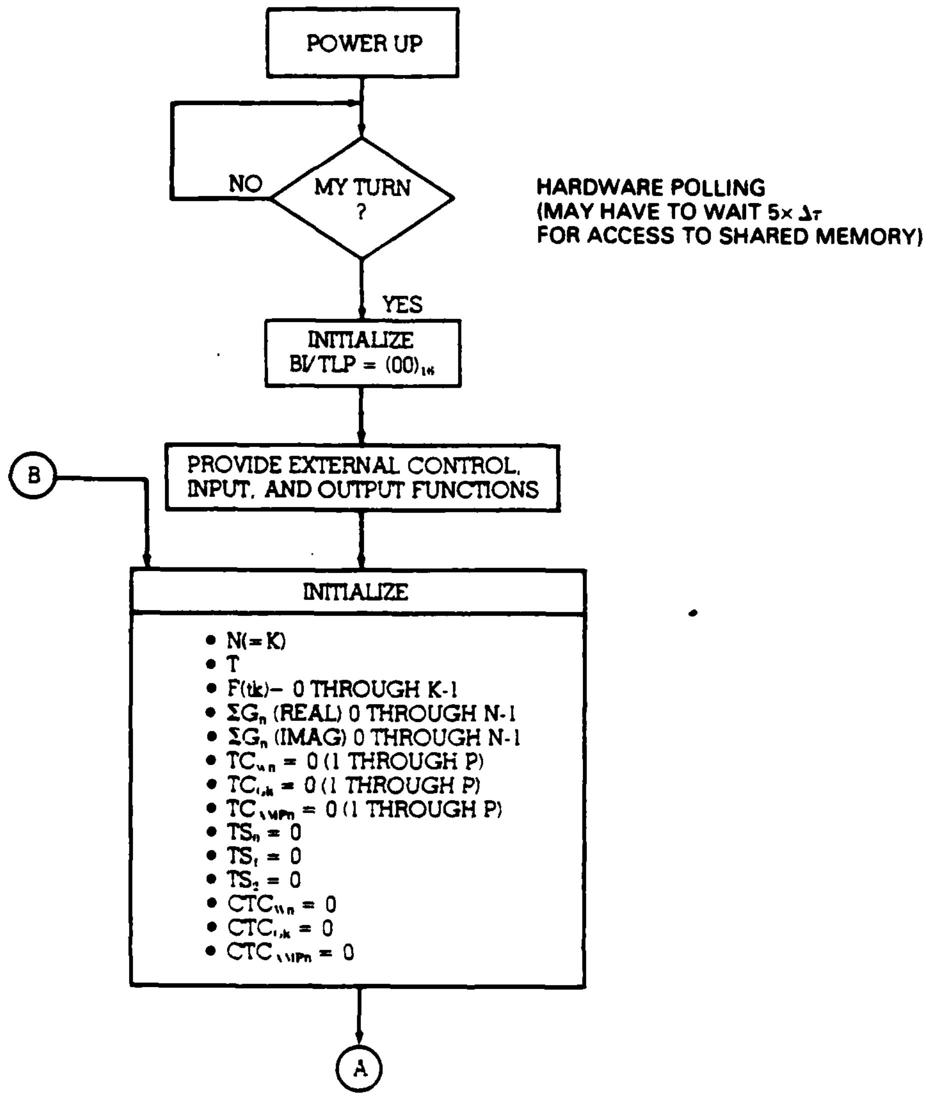

POWER UP
MY TURN ?
NO
YES
HARDWARE POLLING (MAY HAVE TO WAIT 5× Δ_T FOR ACCESS TO SHARED MEMORY)
INITIALIZE BI/TLP = $(00)_{16}$
PROVIDE EXTERNAL CONTROL, INPUT, AND OUTPUT FUNCTIONS
B
INITIALIZE
• N(=K)
• T
• F(tk)– 0 THROUGH K-1
• ΣG_n (REAL) 0 THROUGH N-1
• ΣG_n (IMAG) 0 THROUGH N-1
• TC_{Wn} = 0 (1 THROUGH P)
• $TC_{i,k}$ = 0 (1 THROUGH P)
• TC_{WPn} = 0 (1 THROUGH P)
• TS_n = 0
• TS_t = 0
• TS_2 = 0
• CTC_{Wn} = 0
• $CTC_{i,k}$ = 0
• CTC_{WPn} = 0
A

Figure 11. Master Processor Operational Scenario (Sheet 1 of 2)

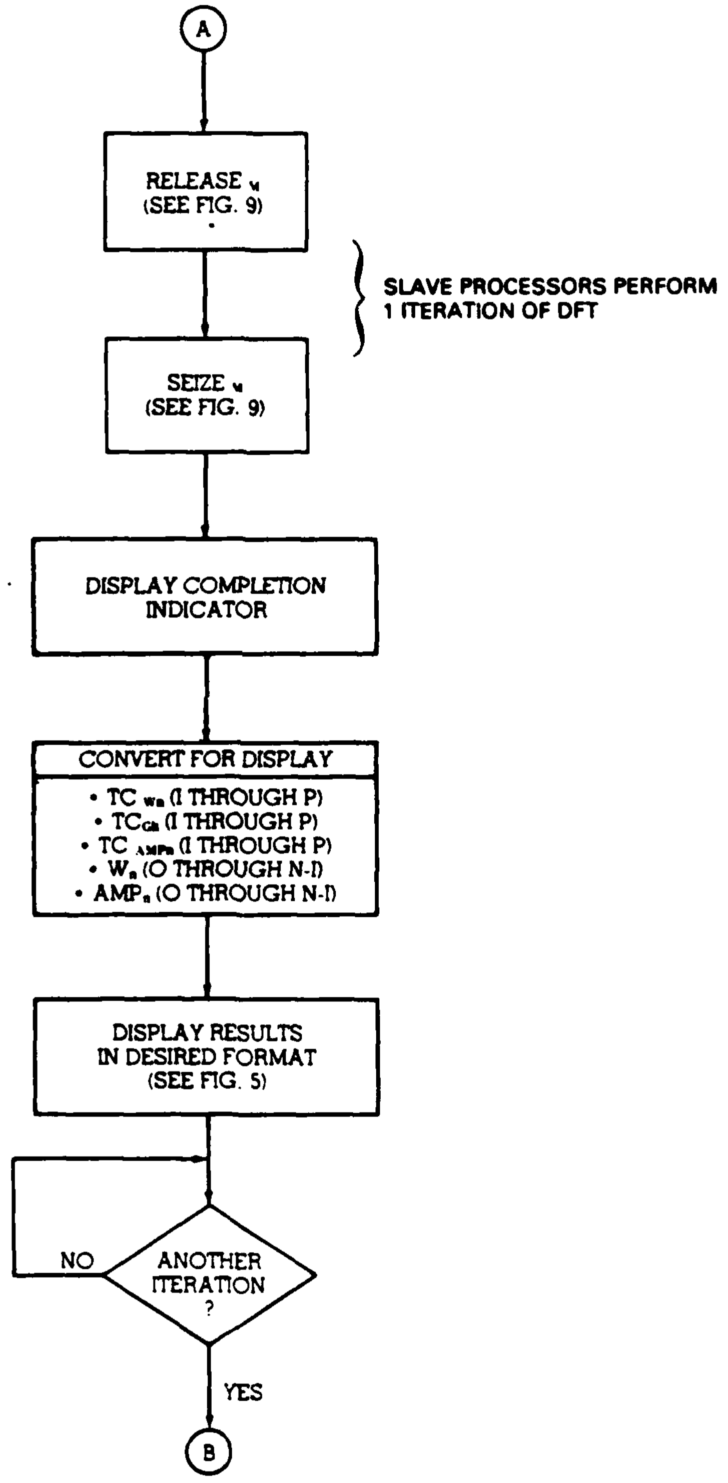


Figure 11. Master Processor Operational Scenario (Sheet 2 of 2)

ASSUMPTIONS:

- Operational programs already loaded into RAM or resident in EPROM
- Slave processor programs stored in local memory
- On power up, slave processors execute time delay software until master processor can initialize BI semaphore

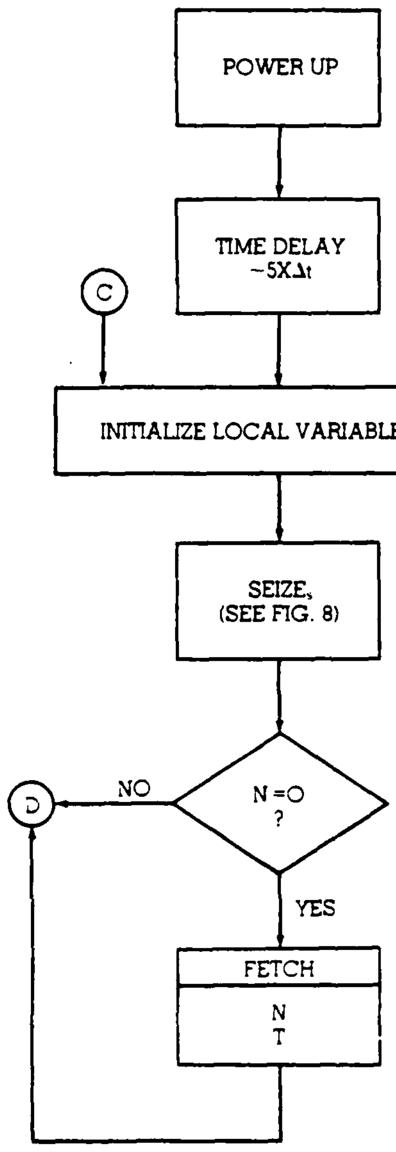


Figure 12. Slave Processors Operational Scenario (Sheet 1 of 4)

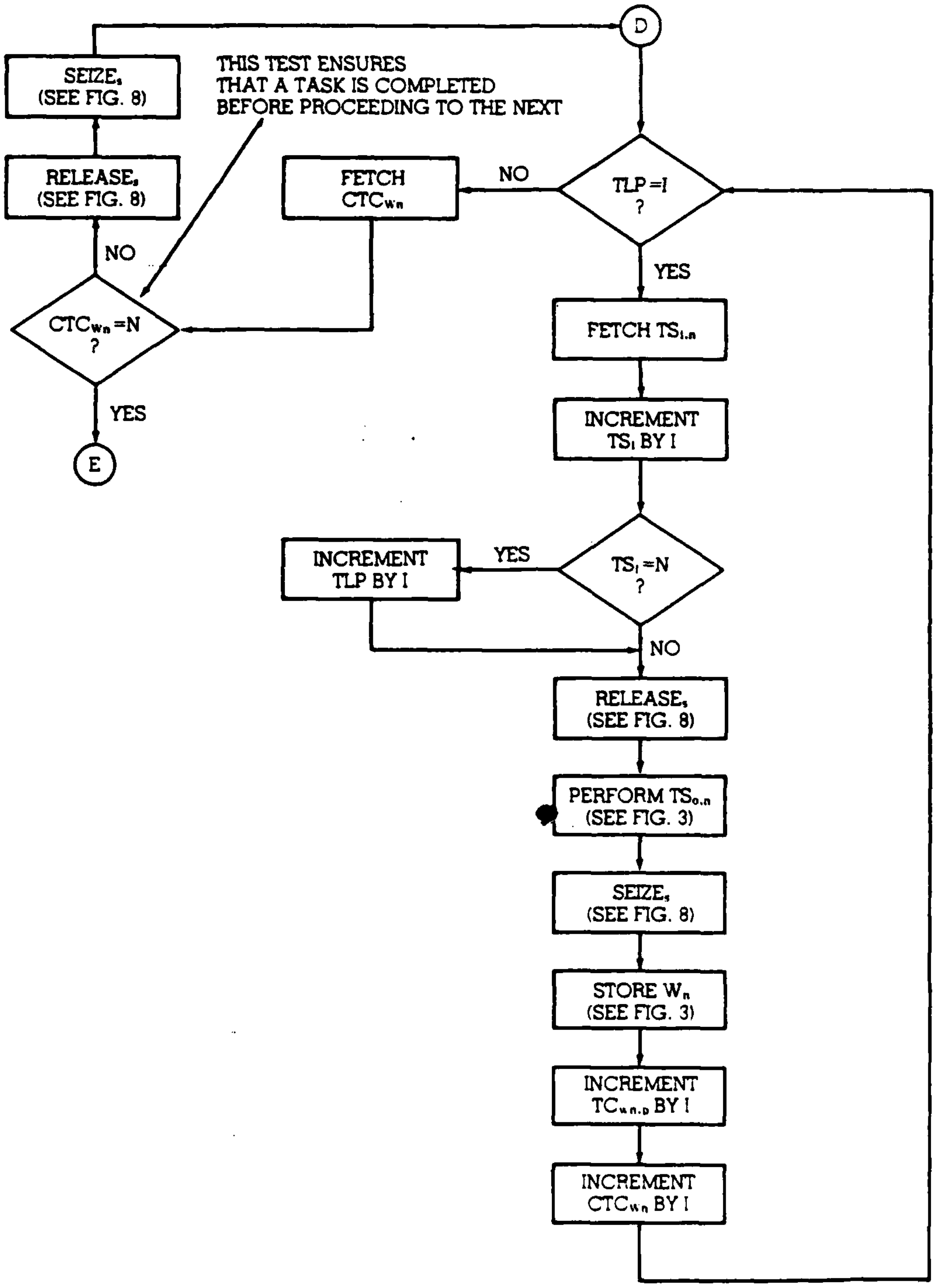


Figure 12. Slave Processors Operational Scenario (Sheet 2 of 4)

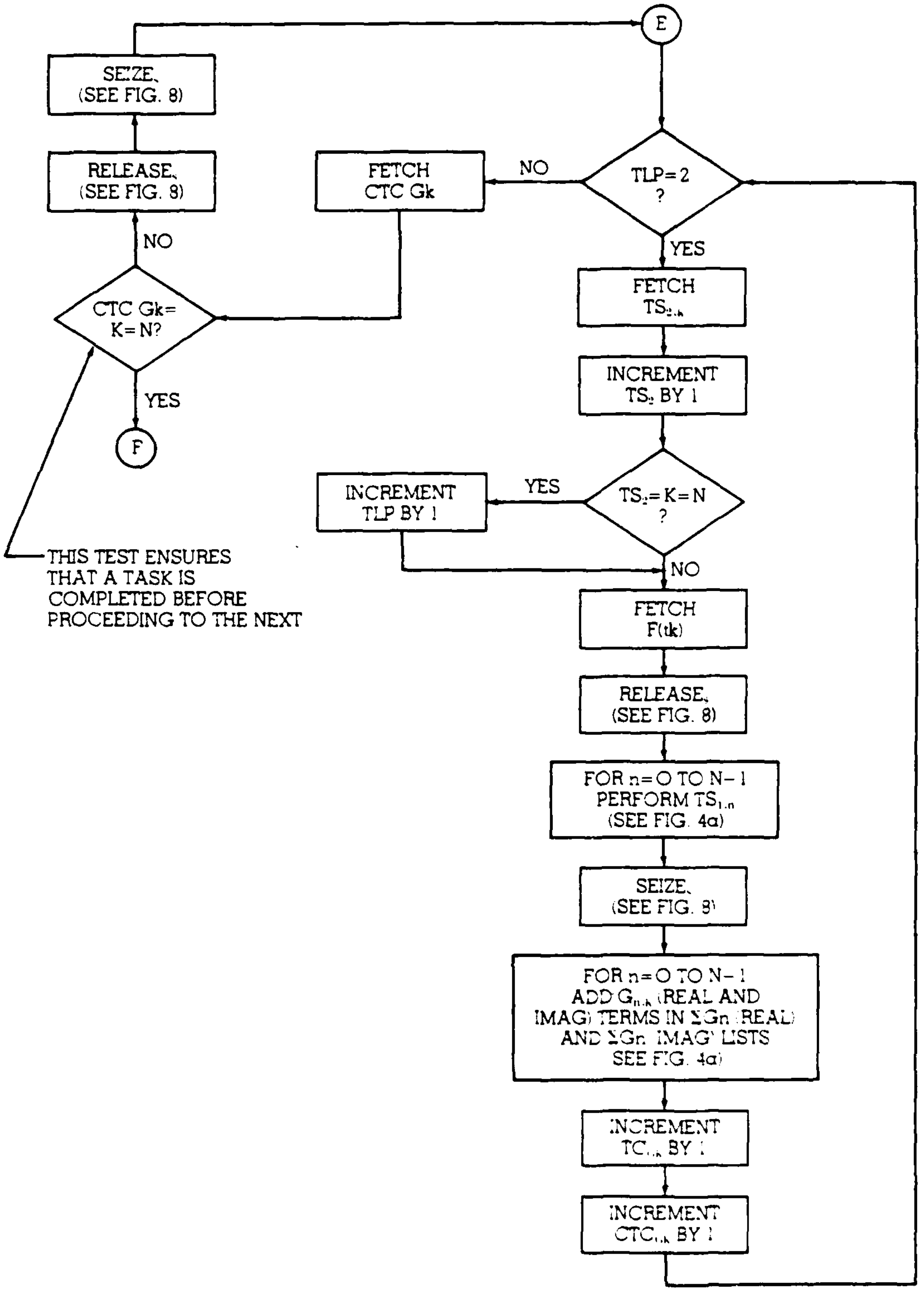


Figure 12. Slave Processors Operational Scenario (Sheet 3 of 4)

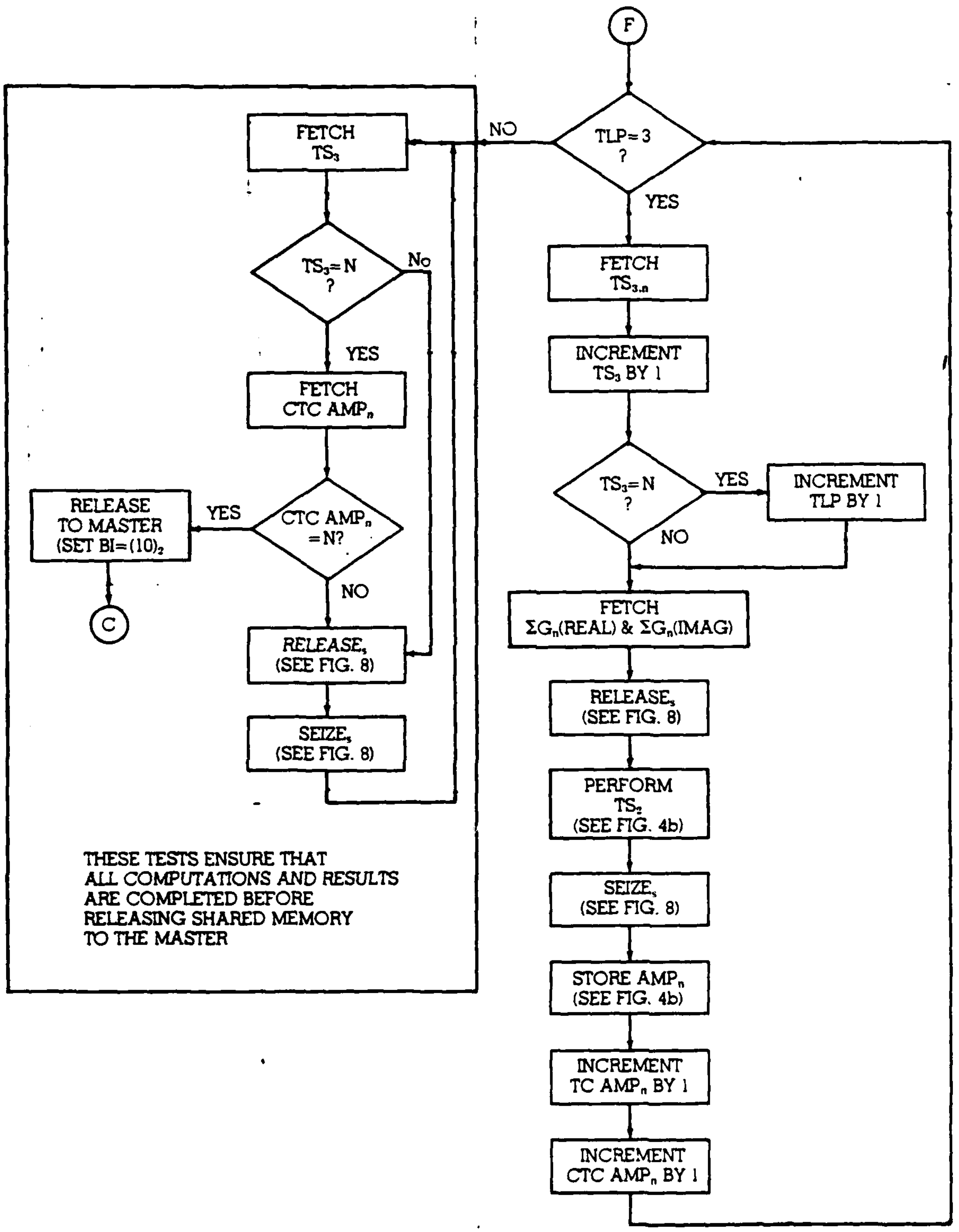


Figure 12. Slave Processors Operational Scenario (Sheet 4 of 4)

REFERENCES

a. Distributed Processing Task, Microprocessor Applications to Avionics FY 80 Plan of 29 Nov 1979

b. Hovanessian, S. A., and Pipes, L. A., "Digital Computer Methods in Engineering," McGraw-Hill Inc., 1969

c. Holt, R. C., et al., "Structured Concurrent Programming with Operating Systems Applications," Addison-Wesley Publishing Co., 1978